TECHNICAL RESEARCH REPORT

Key Establishment in Heterogeneous Self-Organized Networks

by Gelareh Taban, Rei Safavi-Naini

TR 2007-6

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2007		**00-00-2007 to 00-00-2007**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Key Establishment in Heterogeneous Self-Organized Networks	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Maryland,College Park,MD,20742**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**17**	

Key Establishment in Heterogeneous Self-Organized Networks

Gelareh Taban[1] * and Rei Safavi-Naini[2]

[1] University of Maryland, College Park, gelareh@umd.edu
[2] University of Calgary, rei@cpsc.ucalgary.ca

Abstract. Traditional key pre-distribution schemes in sensor and ad hoc networks rely on the existence of a trusted third party to generate and distribute a key pool. The assumption of a single TTP however can be very strong in practice, especially when nodes belong to different domains and they come together in an ad hoc manner. Other important motivations to omit a TTP include preservation of privacy in a network as well as reducing the required knowledge base for the usage of sensor networks. In this work, we show the shortcomings of the previous approaches [3, 13] in terms of both efficiency and security. By incorporating a heterogeneous network, we show that we can dramatically reduce the load on resource constrained devices while also increasing their security. We also propose a new strengthened security model for self-organized ad hoc networks and evaluate the security of our protocol in this model. We evaluate the correctness of the protocol and show that we can achieve network connectivity with very high probability.

1 Introduction

Traditional ad hoc and sensor network settings generally assume a trusted third party (TTP) who is trusted with the keying information and enables secure delivery of keys to the network principals and/or nodes. Security associations, such as authentication of nodes or securing communication channels, are then bootstrapped using this information. In key pre-distribution schemes, the TTP allocates keys to each node prior to deployment either randomly from a key pool [8, 5], or by using a well-defined combinatorial structure such as a t-design [10] that ensures the key subsets allocated to the nodes satisfy certain properties.

However, the assumption of a single TTP can be restrictive in scenarios where the network is self-organized and formed without prior planning. In the following we list some of the immediate applications that require distribution of trust.

1. In disaster response scenarios for example, a network may be formed with members belonging to different administrative domains. Furthermore, it might be impossible to access an outside authority due to the lack of preexisting infrastructure or inability to contact off-site systems [12]. In such life-threatening situations, it is not acceptable to deny data from a legitimate principal that might save someone's life. Therefore in such scenarios, a 'best-effort' security model might be appropriate, making strong guarantees when a single trusted third party can be established and making weaker guarantees when no TTP can be assumed.

2. In combat situations it is essential to allow members of a coalition to join and form collaborative groups. In such dynamic coalitions there is typically no single TTP prior to or during deployment.

3. Existence of a TTP is in immediate conflict with privacy enhancing applications. As sensor and ad hoc testbeds have been deployed, it has become clear that user privacy can be easily compromised as a side effect to seemingly innocuous applications [4]. For example a humidity sensing network can also be used to monitor activity in a room as the human body effectively alters the room humidity. Therefore by removing the presence of an all knowing authority (i.e. the TTP), communication can be made private to the restricted user set.

4. Finally, to allow the wide adoption of sensor and ad hoc networks in everyday scenarios, it is desirable to reduce the required knowledge base of network owners. Customers should be able to purchase a set of nodes that are usable upon purchase without requiring the presence of a network administrator. Therefore the node manufacturer can install public data in the nodes that can bootstrap future security associations.

* Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

In the following we focus on the problem of group key distribution in self-organized ad hoc and sensor networks where no single point of trust exists. A group key allows nodes to securely communicate with each other and participate in collaborative tasks. The dynamic property of the network allow new nodes to join or exiting nodes to leave the group. This is an essential mechanism in applications such as 1 and 2 listed above. We consider heterogeneous networks consisting of two types of nodes: typical low performance sensor nodes and more powerful nodes with more computation and communication resources. It has been recently shown [7, 1] that networks that consist of homogeneous nodes cannot scale well and also have lower performance compared to networks that include a number of more powerful nodes. Introducing more powerful nodes also improves reliability and lifetime of the network [1]. Furthermore [14] showed that pairwise communication security in the presence of a TTP is not necessarily sacrificed if a key distribution scheme leverages the existence of more capable nodes.

1.1 Related Work

The first work on key pre-distribution in ad hoc network without a TTP is due to Chan [3]. In this construction each group member individually selects his keys from a common *public* key pool in a specified way. The aim of the protocol is to probabilistically construct a *Cover Free Family (CFF)* that will ensure shared keys between nodes. After the *key selection phase*, nodes follow a *shared key discovery protocol* that uses homomorphic encryption to discover nodes' shared keys. Chan showed that his proposed protocol allows any two nodes to communicate securely with a high probability and the system provides security against collusion attack. However, [15] showed that the probability that the constructed structure is a CFF, is very low and so the protocol cannot achieve its suggested goal.

The closest work to our scheme (in fact motivation of our work) is Luo et. al [13] which has been inspired by Chan's work. Luo et. al propose a probabilistic group key management protocol (referred to as LSBS) for ad hoc networks and assume homogenous nodes. The objective of LSBS is to establish a common shared key for the whole group. The protocol consists of three phases. In the first step, nodes agree on system parameters and a public key pool. Then each node randomly selects a set of keys from the key pool in accordance with the protocol specification, and performs a shared key discovery (SKD) protocol with each neighboring node to discover shared keys. The group key is generated by special subsets of nodes called initiating groups (IG), and is distributed by flooding the network. Authors show that the success probability of establishing a group key can be made very high if the size of the key ring is chosen appropriately and keys are selected from a structured key pool according to a specified strategy. Authors analyzed the security of the protocol against an eavesdropping adversary who tries to guess the key ring of a node, or the secret key that is used to secure the communication of two nodes.

Shortcomings of LSBS protocol. Although LSBS protocol achieves its stated goal, in practice there are challenges that if not addressed makes the protocol impractical. The following is a list of the more stringent shortcomings of the protocol.
1. LSBS implicitly assumes that a single IG is formed where in practice many IGs may simultaneously exist. In fact our simulation results show that in a network of 1000 nodes, where each node has a key ring of size 150 keys, we can form up to 100 IGs. To obtain a single group key for all nodes some mechanism for negotiation and/or cooperation among IGs is required. Both these approaches[3] however substantially increases the communication and computation cost which is very undesirable in a resource constrained network. The solutions also needs to be carefully designed to prevent security compromise.

2. The communication cost of the shared key discovery (SSD) phase of the protocol is $\mathcal{O}(l)$ where l is the size of the key ring. LSBS requires a node u to execute the SSD protocol with all of its neighboring nodes. If on average a node is in the neighborhood of d other nodes, a communication cost of $\mathcal{O}(d \cdot l)$ per node is incurred. For networks with battery powered nodes it is essential to reduce this cost in order to prolong network lifetime.

3. LSBS is analyzed using a simple threat model that does not take into account real life threats in a wide range of application scenarios. The adversary is considered passive and can only eavesdrop on the communications. Given that the key pool is public, the adversary's objective is to either determine the node key or the link key that secures the link between two nodes.

In sensor networks it is common to assume that the adversary can compromise a subset of nodes and obtain the secret information of the nodes. Such information includes the key rings of the node and the keys that the

[3] For example IGs may negotiate amongst themselves to determine a single IG that is responsible for group key generation. Implementing a fair and democratic negotiation in general would be hard to implement. An alternative approach would be to allow all IGs to contribute partial shares of the group key and distribute these shares to the rest of the network via flooding.

nodes share with their neighbors. This latter information will reduce the effort required for finding the key rings of uncompromised nodes, and/or the link keys for links between the compromised node and its neighbor nodes.

1.2 Our Contribution

In this paper, we propose a Layered Key Pre-Distribution (LKD) Scheme for networks of heterogenous nodes: resource constrained nodes (level 2 or L2) and a small number of high performance nodes (level 1 or L1). L1 nodes have more resources and are possibly better protected (e.g. use tamper proof hardware). LKD uses an unbalanced distribution of keys, where L1 nodes are allocated a larger key ring. The L1-centric clusters that are formed result in more efficient generation of group keys.

We give a probabilistic analysis of the protocol and show that the inclusion of a small number of more powerful nodes in the network results in constant communication and computation cost, independent of the neighborhood size of a node.

We support our evaluation of LSBS (e.g formation of multiple IGs) and our analysis of LKD by simulating a network of 1000 nodes where approximately 6% of the nodes are more powerful. An interesting byproduct of our simulation has been the uncovering of a number of details that must be addressed when the protocol is used in practice. For example, a node may belong to multiple IGs and there must be an efficient decision strategy to participate in a single one. Another example is how to ensure the neighborhood condition is satisfied for all the nodes in an IG.

We next evaluate the security of the protocol in a strengthened security model. We argue that with a public key pool and without a TTP, previous proposed threat models and security metrics such as network resiliency [5, 8], which assumed secret key pool and a TTP, are no longer valid. We update these definitions for our new system and trust model and define a new security metric called neighbor resiliency. We analyze the security of both LKD and LSBS under this new threat model. Our analysis shows that LKD achieves better security than LSBS against node compromising adversaries because sensing nodes in LKD learn much less information about the nodes in their neighborhood.

The paper is organized as follows: Section 2 describes our network and trust model; Section 3 introduces the LKD protocol; Sections 4, 5 provide the correctness and the security analysis of the LKD protocol; Section 6 supports the theoretical analysis with simulation results. We also include theoretical and simulation analysis of LSBS to point out its shortcomings. We provide concluding remarks and future directions in Section 7.

2 System Model

We consider the network to be fully self-organized, meaning that there is no infrastructure (hence no public key infrastructure). Traditional network models considered for sensor models not only assume a homogeneous network but also assume either a grid or a random graph [8, 5] model where all neighboring nodes are in communication contact. A more realistic model takes into consideration the various signal-blocking barriers and interference sources such as hills and buildings that exist in the deployed environment. In practice, deployed nodes are often segregated into exclusive neighborhoods due to the features of the landscape [14]. Our model accounts for this by considering a cluster based network, where sensor nodes form ad hoc groups around more powerful nodes which act as the backbone of the network. Therefore the sensor nodes connect to the rest of the network through the powerful 'gateway' nodes.

We assume a heterogeneous sensor network of size n consisting of two types of nodes: sensing or level 2 (L2) nodes which are resource constrained and have limited storage and energy capabilities and level 1 (L1) nodes which are more capable, with larger memory, more powerful transceivers and energy source. As a result L1 nodes can store larger key rings and other state data as well as communicate with a larger neighborhood of nodes. The network consists of c L1 nodes and $(n-c)$ L2 nodes. Example L2 nodes are small Berkeley Mica2 motes with 8-bit 4MHz processors and 128 KB memories [2]. L1 nodes can be more powerful nodes such as laptops, mobiles or other portable devices. Many such devices have better physical protection against compromise, such as the use of tamper resistance hardware. However for simplicity, we assume the same type of protection for L1 and L2 nodes. We also assume that each node u_i has a unique identifier i.

Trust Model. We assume that the network has no central authority or a single trusted third party. Each node essentially acts as its own domain authority. Public information such as the key pool is available to all network parties, including malicious parties.

Authentication. Since we do not assume any trusted third parties, it is impossible to establish strong authentication and identification amongst network nodes. We weaken our requirements such that to control the join of

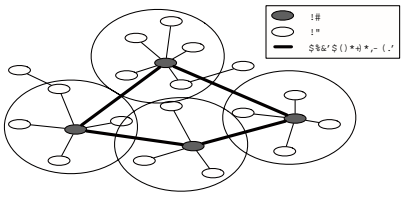


Fig. 1. A heterogenous sensor network with two types of nodes, L1 and L2.

malicious nodes to the group, we assume some auxiliary identification mechanism for nodes (e.g. node hardware). Details of such a mechanism is outside the realm of our work.

3 Layered Key Pre-Distribution (LKD) Scheme

In this section we describe the LKD scheme to establish both pairwise and group keys in a self-organized network that does not have a TTP. The heterogenous network consists of resource constrained nodes (L2) and more capable nodes (L1) that contain a larger portion of the key pool than L2 nodes. It follows that L1 nodes are able to establish secure links with a larger portion of the nodes. In each neighborhood, local (l, r)-secure groups are established where l denotes the security level and r is the minimum number of nodes in the group. We will show later that r does not effect the security of the protocol and is used for efficiency purposes. Local groups in a neighborhood together generate a cluster group key which are exchanged to contributively generate a network group key. We ensure that the key generated in each layer (i.e. local, cluster or network) is independent. The overall algorithm consists of the following phases: initial setup, neighborhood discovery, cluster and group key generation, join and leave. Figure 2 the outline of the steps of the protocol.

Initial Setup. Nodes agree on network parameters and select keys rings.

Neighborhood Discovery. Nodes discover the node types in their neighborhood. If an L2 node discovers an L1 node as a neighbor, it executes the shared key discovery phase which consists of a private set intersection protocol. A secure link can be established between the L1 and L2 nodes based on their shared keys. L1 nodes keep an account of all the L2 nodes in their neighborhood by computing an incidence matrix.

Cluster and Group Key Generation. L1 nodes use the incidence matrix to assist their neighboring nodes to form local (l, r)-secure groups. Each local group contributively generates a partial cluster and group key. The cluster and group keys are thus generated democratically by a large portion of the neighborhood nodes.

Join. A newly deployed node joins the network.

Leave. A possibly malicious node departs from the network.

Fig. 2. Outline of LKD protocol.

3.1 Initial Setup

In this phase nodes agree on parameters used in the protocol. The system parameters include a public key pool and its partition into κ blocks of size m each. The security parameter is l which defines the level of link security by specifying the minimum number of keys two nodes need to share to establish a secure communication channel. The size of the key rings of L1 and L2 nodes are also set to k_A and k_B.

We note that these parameters can either be set by the node manufacturers or during an initial setup phase prior to deployment.

Distributed Key Ring Selection: A node u_i randomly selects one key from each key block to form a key ring $\{K_1^i, K_2^i, ..., K_k^i\}$, where $k = k_A$ for an L1 node and $k = k_B$ for an L2 node. Assume that k_B is equal to the number of blocks in the key pool, κ. Since $k_A > k_B$, an L1 node needs to choose more than one key from each block. We define the following strategy for key ring selection of L1 nodes.

L1 node Key Selection:

– Let $k_A = tk_B + s$, where $t, s \in \mathbb{Z}$. Select t keys from block 1 to $(k - s)$.
– Select $(t + 1)$ keys from blocks $(k - s) + 1$ to block k (in total s key blocks).

3.2 Neighborhood Discovery Phase

In this phase, L1 nodes initially send beacons identifying themselves as 'L1' nodes to their neighborhood nodes. The beacon message for L1 node u_i can take the simple syntax of $< i, L1 >$ where i is the node identifier.

An L2 node 'discovers' an L1 node when it hears its beacon message. To establish a secure channel with the L1 and help populate L1's incidence matrix, it runs a secure shared key discovery (SSKD) protocol, reminiscent of [3, 13]. This SSKD protocol is essentially a privacy preserving set intersection protocol that allows the two participating parties to discover their shared keys from their individual key sets.

For L1 node v_i, the incidence matrix I^i has k columns labeled by the node keys $K_1^i, \cdots K_k^i$, and one row for each neighbor. $I^i(j, t) = 1$ if K_t^i is shared with node u_j in the neighborhood of v_i, and zero otherwise. The incidence matrix of v_i can be used to keep an account of the keys shared by the nodes in L1's neighborhood, *given that the keys are shared with* v_i. This property is important as it maintains the optimal privacy for the neighboring L2 nodes. Specifically v_i does not learn any information about the key ring of its neighboring nodes other than the shared key information it learns during the execution of the SSKD protocol.

If an L2 node is not directly connected to an L1 node (i.e. it is isolated from an L2), it simply waits and performs the join protocol after the key establishment protocol is complete.

In this step, L1 nodes also discover each other and establish an l-secure channel between pairs of nodes. This communication network forms the backbone of the larger network.

Secure Shared Key Discovery (SSKD) Consider the case when node u_j wants to discover the keys it shares with node u_i. Let u_i have keys $K_1^i, K_2^i, ..., K_l^i$ and u_j have $K_1^j, ..., K_m^j$, where $l, m \in \mathbb{Z}$. Assume the existence of a homomorphic encryption scheme, where $E_k(m)$ denotes encrypting message m using key k. The SSD protocol is as follows:

1. u_i forms polynomial $f_i(x) = (x - K_1^i)...(x - K_l^i)$ and send to u_j the encrypted coefficients, $E_{K_i}(\cdot)$.
2. u_j computes $z_g = E_{K_i}(rf_i(K_g^j))$ using the homomorphic property of the encryption scheme, where r is a random number. u_j returns z_g to u_i.
3. u_i decrypts z_g to obtain $rf_i(K_g^j)$. If the value is zero, then they have a common key.
4. u_i returns to u_j an m-bit bitmap with 1 at bits where $rf_i(K_g^j) = 0$ and 0 elsewhere.

In contrast to LSBS, our SSKD protocol requires the nodes to exchange an m-bit bitmap indicating the shared keys of the participating nodes (step 4). The main reason for this inclusion is that unlike LSBS, in our protocol, L1 nodes can select more than one key from each block. Therefore nodes must indicate which key in the block is shared or not, using the bitmap. LSBS considers a homogeneous network where every node picks one key from each key block.

Securing Bitmap Transmission A potential security leakage is the bitmap exchange step of the SSD, which identifies to an eavesdropper the number of shared keys of two nodes. This can aid a smart adversary to compromise a neighboring node which shares the most keys with a target node, as well as reducing the search space for the channel securing key.

The following protocol takes advantage of the privacy preserving characteristics of a homomorphic encryption scheme such as El Gamal [9]. We show that although El Gamal is a public key encryption scheme, our scheme does not require the public key infrastructure (which inherently assumes a TTP) or the computationally expensive computations generally associated with public key schemes. Therefore its use in our protocol is practical.

Assume node u_i wants to privately send a k-bit bitmap b to node u_j. We use the *multiplicative homomorphic properties* of the El Gamal [9] encryption scheme for u_i to send b to u_j. Specifically this property is defined as: $E_K(m_1 m_2) = E_K(m_1) \times E_K(m_2)$ where $E_K(m)$ is the encryption of m using key K.

Let the El Gamal public key of u_j be (g, h) and the secret key be $(x = log_g h)$.

$\mathbf{u_j} \rightarrow \mathbf{u_i}$: $r, d \leftarrow \{0,1\}^*$; Send $< C_1, C_2 >=< g^r, h^r \cdot d >$, (g, h)
$\mathbf{u_i} \rightarrow \mathbf{u_j}$: $r' \leftarrow \{0,1\}^*$; Send $< C_3, C_4 >=< C_1 g^{r'}, C_2 h^{r'} \cdot m >$
$\mathbf{u_j}$: bitmap $b = \frac{C_4}{C_3^x \cdot d}$

Node u_j encrypts a dummy message d and sends to u_i the ciphertext and its public key. u_i multiplies the bitmap with the ciphertext and randomizes the message using r'. Using its private key u_j can decrypt the processed ciphertext and obtain the bitmap. This protocol ensures that the bitmap remains private to u_i, u_j assuming the El Gamal encryption scheme is secure.

By loading nodes with a set of random r values and associated g^r, h^r during the setup phase it is possible to reduce the amount of computation needed to simply one exponentiation and two multiplications per node. Furthermore we note that although we are using public key cryptography, we do not rely on the existence of a PKI and therefore we preserve the distributed nature of the network. Finally, we point out that this step is only performed once or twice by sensing nodes through out their lifetime. In fact by using the following strategy we can reduce this step to be used only when necessary:

Strategy: *Initiate this protocol if and only if there are shared keys. If there are no shared keys, simply send a NULL message.*

We emphasize that using the El-Gamal protocol to secure bitmap transmission is an optional step that can still be omitted in order to conserve energy. That is, we trade security for efficiency. An optional symmetric key protocol which achieves some measure of security as long as the adversary has not compromised any nodes is to use a global secret key to privately transmit the bitmap.

3.3 Cluster and Group Key Generation

In this phase, L1 nodes v_i use their incidence matrix I^i to assist the nodes in their neighborhoods to initiate local (l, r) groups where a minimum of r nodes share l keys. This is done by finding a set of r rows $\mathcal{R}$ and at least l columns $\mathcal{L}$ in the incidence matrix for which an (l, r)-secure subset can be formed. The formation of the local groups allow v_i to communicate to a group of nodes via multicast thus reducing communication. Also nodes in local groups contribute to the formation of the cluster keys thus preventing the selection of weak keys.

Once this local (l, r) group is formed, v_i informs the group members of their group membership using secure channels. Local group members now can communicate securely using their secret group key K_L, where $K_L = K_1^L \oplus ... \oplus K_l^L$ where $\{K_1^L, ..., K_l^L\}$ are the set of shared keys in the local group L. Each local group L contributively generates a partial cluster key K_C^L in order to democratically agree on a cluster key K_C. We note that potentially two L2 nodes which are not in direct communication can belong to the same local group. This is because they have a smaller transmission range than the L1 nodes. In this case, the L1 node can be used as an intermediate routing point to forward messages. It is also possible to reduce this form of routing if we assume a *directed antenna* for the L1 nodes. Then the L1 node can group an (l, r) subset together if and only if they are in the same vicinity.

Cluster Key Generation. We require that all members of the local group L contribute in the generation of the partial cluster key K_C^L. To form a partial cluster key nodes in L:

1. For all $u_i \in L$:
   - u_i randomly selects its key share s_i;
   - u_i encrypts $E_{K_L}(s_i)$ and broadcasts to L;
2. For all $u_j \in L, j \neq i$, u_i decrypts $D_{K_L}(s_j)$. The partial cluster key by group L is calculated as $K_C^L = s_1 \oplus ... \oplus s_c$.

We note that L1 node v_i can overhear all communications in its neighborhood since it also shares the local group key. Once all partial cluster keys are generated, v_i computes the final cluster key $K_C = K_C^{L_1} \oplus ... \oplus K_C^{L_x}$ where $\{ L_1, ..., L_x \}$ are the set of local groups formed in the neighborhood. v_i can then transmit the final cluster key to its neighborhood nodes using the secure local group keys.

Group Key Generation. The group key can be generated similar to the cluster key by requiring nodes to select a key share for the group key along with the cluster key share. L1 nodes then exchange the partial group key generated in their neighborhoods to arrive at the final group key.

3.4 Join

A newly deployed node u_i can join the network by establishing an l-secure channel to a node u_j which already belongs to the secure group. u_j essentially acts for u_i as the 'gateway' to the network. To achieve forward security, the cluster and group key of the cluster and whole group respectively, is renewed by applying a one way function to the current session key. u_j then forwards the new session key to u_i using the secure channel. The nodes use the previously described SSKD protocol to discover at least l shared keys and establish a secure channel. If they cannot do so, u_i contacts other nodes in its neighborhood. As a result the new node only knows the keys it shares with the 'gateway' node as well as any previous nodes it had contacted prior to establishing the secure channel.

Similarly an L2 node which is isolated from the neighborhood L1 node due to either being out of the range of the L1 node or by not being to establish a secure channel with L1 in the key establishment phase, can join the group using the above protocol.

3.5 Leave and Node Revocation

If an L2 node u_i decides to leave the group, the neighborhood L1 node v_i can use its incidence matrix to determine the keys that a departing node has in common with the other nodes in its neighborhood and if need be, purge these keys. It then alerts the nodes in L, the local group of u_i, to also purge their key rings. As a result, the departing node no longer has any information regarding the key rings of the nodes in its neighborhood.

If the departing node is malicious, the cluster may decide to not only purge the keys but also to compute a new cluster and group key. As such, v_i randomly selects another local group L' in its neighborhood and requests the nodes in L' to re-execute the cluster key generation protocol to generate $K_C^{L'}$. The node v_i then recomputes the cluster and group keys and securely transmits them to the effected nodes. The new cluster and group keys are independent of the old keys since $K_C^{L'}$ is a random value.

Note that the existence of the local groups allows the re-generation of the cluster and group key to be very efficient. Furthermore by randomly selecting the local group, v_i distributes the added computation and communication load uniformly amongst the nodes in its neighborhood.

4 Correctness Analysis

In this section we show the correctness of the LKD protocol. We say that LKD is *correct* if the protocol allows the 'backbone' L1-network as well as the cluster of L2 nodes around an L1 node, to be connected and thus functioning with a high probability. Later we verify our results by simulation. In the next section we analyze the security of the protocol against both a passive and an active adversary.

In our theoretical analysis we limit the key ring size of L1 nodes $k_A = t \cdot k_B + s$ as follows (k_B is the key ring size of L2 nodes): $t = 1$, $s = [0..k_B]$. In the following, we will first analyze the case where $s = k_B$ and then when s can be assigned any value from $[0..k_B]$. To establish an l-secure link, two nodes share at least l keys.

For readability purposes, in the rest of the paper we use the notation A and B to refer to L1 and L2 nodes respectively.

4.1 Case 1: $k_A = 2k_B$, where $s = k_B$

We have the following proposition, with proof provided in Appendix A.

Proposition 1. *Let $P_B(r,l)$ denote the probability that r L2 nodes share l keys, $P_{A,B}(r,l)$ denote the probability that a group of $(r-1)$ L2 nodes and one L1 node share at least l keys and $P_{A,A}(2,l)$ denote the probability of two L1 nodes sharing l keys. Then, we have the following.*

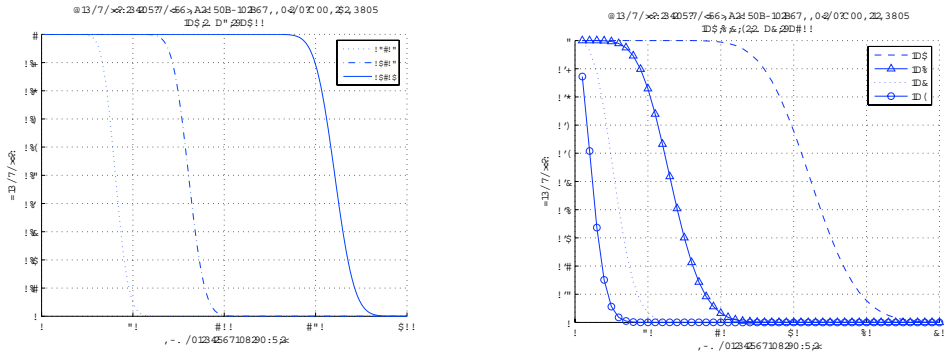
$$P_B(r,l) = \binom{k}{l} \; m^{k(1-r)}(m^{r-1}-1)^{k-l} \tag{1}$$

$$P_{A,B}(r,l) = \binom{k}{l} \, 2^l m^{k(1-r)}(m^{r-1}-2)^{k-l} \tag{2}$$

$$P_{A,A}(2,l) = \left(\frac{(m-2)(m-3)}{m(m-1)}\right)^k \sum_{\substack{\alpha,\beta \\ 2\alpha+\beta=l}} \binom{k}{\alpha}\binom{k-\alpha}{\beta} 2^{\alpha+2\beta}(m-2)^{-\alpha}(m-3)^{-(\alpha+\beta)} \tag{3}$$

The probabilities above are derived by finding the probability that two nodes share a key in a key block and then adding these independent events to obtain the appropriate binomial coefficients. The main difference between $P_B(r,l)$ and $P_{A,B}(r,l)$ is the probability of sharing a key in a block, which changes from $\frac{1}{m}$ to $\frac{2}{m}$. To find $P_{A,A}(2,l)$, we need to consider the case when nodes share 0, 1 or 2 keys in a key block. In the above formula, α and β represent blocks that share 2 and 1 keys respectively. We need not consider blocks that do not contribute any keys.

Figure 3(a) graphs the obtained probability equations, comparing the probabilities of two nodes establishing an l-secure channel for different node types, when the key pool is made up of 200 blocks, with a block size of five keys. We can see a rapid transition in the probability of establishing an l-secure channel for different l. For example,



(a) Two nodes (b) $(r-1)$ L2 nodes and one L1 node

Fig. 3. Probability of (a) two nodes and (b) r nodes, establishing an l-secure channel, where $k_A = 2k_B$.

as can be seen in the graph, once l approaches 30 for two L2 nodes, the probability of establishing a secure link rapidly drops off from 1 to 0.

Figure 3(b) generalizes the node pair to groups of r nodes. It is intuitive that establishing an l-secure channel becomes less probable as the group size increases. We also note that when there is a high probability for l-secure channel among r nodes, the probability of establishing a secure channel between two L1 nodes will be an even higher value. It is also interesting to note that the phase transition becomes slower as the number of nodes in the group increases.

4.2 Case 2: $k_A = k_B + s$, where $s \in [0, k_B]$

Let set S consist of the s key blocks from which an L1 node selects two keys and let $\bar{S}$ consist of the remaining $k - s$ key blocks.

Let $P_{A,B}(r, l)$ be the probability of r nodes (one L1 node and $(r-1)$ L2 nodes) sharing at least l keys. Let Z_x be the event that r nodes share a key in a given block x. The probability that Z_x occurs, is equal to p_s for blocks $x \in S$, and $p_{\bar{s}}$ for $x \in \bar{S}$. Key collisions for each block can be modeled as independent Bernoulli trials. The generating function for probabilities $P_{A,B}(r, l)$ is calculated as the product of two binomials with success probabilities of p_s and $p_{\bar{s}}$:

$$f(x) = (p_s x + (1 - p_s))^s (p_{\bar{s}} x + (1 - p_{\bar{s}}))^{k-s} \tag{4}$$

Proposition 2. *The probability that the r nodes share exactly l keys is equal to the coefficient C_l of the x^l term in polynomial equation 4, and*

$$P_{A,B}(r, l) = \sum_{i=l}^{k_B} C_i \tag{5}$$

where C_i is the coefficient of the x^i term in $f(x)$ and k_B denotes the size of the key ring of L2 nodes.[4]

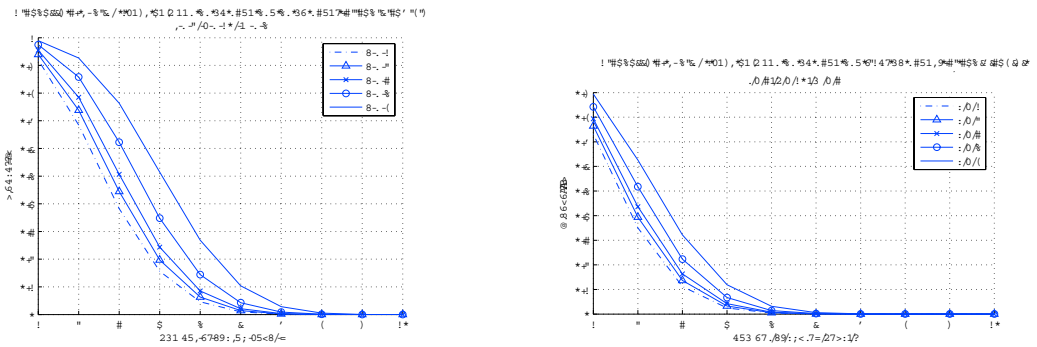
Proposition 3. *Let $P_{A,A}(2, l)$ be the probability of two L1 nodes sharing at least l keys. Let α, β, γ be non-negative integers satisfying $2\alpha + \beta + \gamma = l$.*

$$P_{A,A}(2, l) = \sum_{\substack{\alpha,\beta,\gamma \\ 2\alpha+\beta+\gamma=l}} \binom{s}{\alpha}\binom{s-\alpha}{\beta}\, p_2^{\alpha}\, p_1^{\beta}\, p_0^{s-\alpha-\beta} + \binom{k-s}{\gamma}\, \tilde{p}_1^{\gamma}\, \tilde{p}_0^{k-s-\gamma} \tag{6}$$

where p_i is the probability of sharing i keys for the first s blocks and $\tilde{p}_i$ is the probability of sharing i keys for the remaining $k - s$ blocks.

This proposition is based on the fact that the first s blocks can contribute 0, 1 or 2 shared keys per block, and the last $(k - s)$ blocks can contribute 0 or 1 shared keys per block. In the above formulae, α represents blocks that share 2 keys and β and γ represent blocks that share only 1 key in S and $\bar{S}$ respectively.

[4] Examples to illustrate how the above proposition can be used are provided in Appendix B.



(a) Two nodes

(b) $(r-1)$ L2 nodes and one L1 node

Fig. 4. Probability of (a) two nodes and (b) r nodes, establishing an l-secure channel, where $k_A = 2k_B + s$.

Figure 4(a) graphs the probability of establishing an l-secure channel between an L1 node and an L2 node for different values of s. The results confirm intuition by showing that as the key ring of an L1 node becomes larger, the probability of a secure connection with a L2 node increases. A similar result is verified in Figure 4(b) when we consider r nodes, consisting of one L1 node and $(r-1)$ L2 nodes.

In a more general version of this problem, a node can select extra keys from any block of its choosing, rather than the first s blocks. It is intuitive that in this version of the problem, the probabilities of establishing an l-secure channel do not increase to the same extent as the more special case presented above. We leave the analysis of this problem as a future exercise.

The graphs presented in this section, allow a network administrator to choose appropriate values for the system parameters. In the following section, we show how an increased key ring not only increases the probability of establishing a secure channel (as shown), but also decreases the security of the system. It is therefore important to achieve the proper balance between connectivity and security. Section 6 gives simulation results to confirm the presented theoretical results.

5 Security Model and Analysis

5.1 Adversary Model

We analyze the security of LKD against two types of adversaries.

1. *Passive Adversary (PA)* with only access to public data (key pool), description of the protocol and transcript of node communications.
2. *Node Capturing Adversary (NCA)* with access to all the information available to a passive adversary, and also the private data of t nodes that it has captured.

Note that we do not allow an NCA adversary to interact with the nodes. That is we only consider the case when the adversary uses its information to eavesdrop on others' communication. The goal of both adversaries therefore, is to learn the secret keys between nodes that are used to secure their links.

5.2 Security Model in Key Pre-distribution Systems

The security of traditional key pre-distribution schemes that assume the existence of a TTP [8, 5, 6, 11], are based on the facts that (i) the keys in the key pool are exclusively secret to the TTP, (ii) nodes key ring are private, and (iii) the link communication is confidential.

In this model an adversary cannot introduce a 'new' device into the network because even if there is no authentication mechanism, it does not have access to the key pool. However by compromising legitimate nodes and obtaining their key rings and/or identities, an adversary can gain entrance into the secure network. The more nodes an adversary compromises, the more it learns of the key pool and the more effective an attack it can launch against a target secure channel. This notion is captured by the *resiliency* of the protocol against node compromise, where resiliency metric is defined to be "the fraction of links in the network a node-compromising adversary is able

to eavesdrop on, as a result of recovering keys from captured nodes"[5]. A protocol has stronger security if the adversary is forced to compromise a larger percentage of the nodes to eavesdrop on a target channel.

Also, in [8, 5] information that an NCA obtains from captured devices combined with the key indices allows him to gain information about the keys belonging to other network nodes.

5.3 Security Model in Self-Organizing Networks

The security of the SO protocols (such as LKD and LSBS) does not rest on the secrecy of the key pool; in fact, the key pool is considered to be public information and can be accessed by the adversary. This means that if there are no auxiliary means of authentication, the adversary can introduce a malicious node v with the aim of extracting key information from a victim node u: v can choose a key ring and run SKD with u to find out a subset of keys of u (that they share). It then can select a new key ring and repeat the protocol. After sufficient runs of this, v can learn all the keys of u. This means that it is crucial to assume a method of node authentication that prevents the adversary from introducing nodes of its choice. Since this is not the focus of our paper, we do not consider this scenario and leave it for future work.

The security of the SO protocols is based exclusively on (i) the size of the key pool and (ii) the security of link keys. In LKD, an NCA gains only local information from a compromised node; that is, it learns only the key ring of the node and potentially any information it shares with nodes it associates with. In the case of LKD, a node u_i associates only with its neighboring nodes $\mathcal{N}_i$, and by compromising u_i an adversary learns not only the key ring of u_i but also the keys it shares with its neighboring nodes. Therefore by compromising u_i, the adversary can tighten its search space when attacking (i) a link between two nodes where at least one is neighbor to u_i or (ii) the key ring of a node neighbor to u_i. We capture this notion in the following security parameter for the SO model: *Neighbor resiliency* is defined as the fraction of the key pool the adversary can discard in its exhaustive key search to attack a target secure channel, as a result of recovering keys from neighboring captured nodes. Another security metric we consider is the advantage the adversary gains in determining the key ring of a node when it is in the neighborhood of a compromised node.

In the following, we analyze LKD against first a passive adversary and then a node capturing adversary.

5.4 Analysis of Passive Adversary

An eavesdropping adversary cannot obtain any information about the keys, except to exhaustively guess at the final shared key between nodes. This is because in the course of the key establishment protocol, no information about the key ring of the nodes is leaked. The adversary knows that there are $N = mk$ possible keys and at least l keys from k different possible blocks are used to secure a link. Therefore, the search space for the attacker is equal to:

$$\sum_{t=l}^{k} \binom{k}{t} m^l \tag{7}$$

Similarly, to determine the key ring of a node of size k, the adversary must exhaustively search $\binom{k}{t}m^l$ possibilities.

5.5 Analysis of Node Capturing Adversary in LSBS

A node capturing adversary obtains not only the node's key ring but also the incidence matrix that contains key information about its neighboring nodes. In particular, it learns how many keys are shared between the neighbors and whether the captured node shares any of these keys on its key ring.

Consider three nodes u_i, u_j and u_c. Assume $u_i \in \mathcal{N}_c$, $u_i \in \mathcal{N}_j$, and u_c is a compromised node. Let k be the size of the key rings of u_c, u_i, u_j respectively. The goal of the adversary is to break the secret link between u_i, u_j.

Case 1: $u_c \notin \mathcal{N}_j$

By compromising u_c, the adversary obtained the following information: u_c and u_i share b keys and do not share $(k_c - b)$. To guess the key ring of u_i, the adversaries' search space is reduced from m^k to m^{k-b}.

The search space to exhaustively guess l shared keys between u_i and u_j is reduced from $\binom{k}{l}m^l$ to $\sum_{\alpha=0}^{l}\binom{k-b}{\alpha}\binom{b}{l-\alpha}m^\alpha$. We can easily see that the search space has been reduced because:

$$\sum_{\alpha=0}^{l}\binom{k-b}{\alpha}\binom{b}{l-\alpha}m^\alpha \leq \sum_{\alpha=0}^{l}\binom{k-b}{\alpha}\binom{b}{l-\alpha}m^l = \binom{k}{l}m^l$$

Therefore the search space to break an l-secure link between u_i and u_j is equal to:

$$\sum_{t=l}^{k}\sum_{\alpha=0}^{t}\binom{k-b}{\alpha}\binom{b}{t-\alpha}m^{\alpha} \tag{8}$$

The neighbor-compromise resiliency can be obtained from Equations 7 and 8.

Case 2: $u_c \in \mathcal{N}_j$
The reduction in the search space of the adversary is even more significant when the adversary can compromise a node in the neighborhood of both u_i and u_j. In this case, the incidence matrix stored in u_c leaks how many keys u_i and u_j share as well as the keys u_c shares with either of these two nodes. Let b_i (or b_j) be the number of keys u_c shares with u_i (or u_j) and $b_{i,j}$ is the number of keys u_c has in common with both u_i and u_j. Let ℓ be the number of keys u_i and u_j share, where $\ell \geq l$ (since u_i and u_j can establish an l-secure channel). Therefore, the search space to break the security of an l-secure channel, has been reduced from Equation 7 to, where $b = b_i + b_j - b_{i,j}$:

$$\sum_{x=\ell}^{k}\sum_{\alpha=0}^{x}\binom{k-b}{\alpha}\binom{b}{x-\alpha}m^{\alpha} \tag{9}$$

5.6 Analysis of Node Capturing Adversary in LKD

In contrast to LSBS, sensing nodes in LKD do not compute an incidence matrix. As a result a compromised L2 node u_c in LKD does not leak any keying information about the nodes in the neighborhood of u_c. Below we itemize the information an adversary learns by compromising u_c:

- The keys that u_c has in common with the L1 node in its cluster, or if it is not connected to an L1 node, the connecting L2 node.
- If it is part of an (l, r)-secure local group, only the keys it shares with *all* of them.

In both cases, the derived metrics are identical to the probabilities of Case 1 in Section 5.5. However the number of links and nodes to which these reduced probabilities can be applied to has been decreased dramatically. This is primarily because LKD does not require an L2 node to connect to every node in its neighborhood. Instead the number of secure connections an L2 node needs to establish as well as the keys it shares with neighboring nodes has been reduced to only those that are necessary.

In the event that an adversary compromises an L1 node and the L1 node does not have any tamper resistant hardware, the adversary gains keying information about all the nodes in its neighborhood. In this case the adversary gains as much information as in the LSBS protocol.

Since the majority of the nodes in the network are L2 nodes, we can conclude that on average the advantage that an adversary gains by compromising nodes in LKD has been reduced and therefore LKD is more secure than LSBS.

6 Simulation and Discussion

In this section, we use simulations to first highlight the shortcomings of LSBS in a practical setting, that is, the large number of IG that are formed and the high communication cost that is incurred. We then show the correctness of LKD protocol by examining the established connectivity against the protocol parameters. We conclude by showing the improved efficiency of the scheme.

6.1 Network Architecture and Setup

The simulation assumes a static network of $n = 1060$ nodes, consisting of 60 L1 nodes and 1000 L2 nodes. This is a reasonable assumption in a dense static network or a highly dynamic network when nodes move around but in a bounded region (e.g a group of rescuers in an emergency situation or troops in a battlefield).

We assume that L1 nodes have twice the transmission range R_A of L2 nodes R_B. To guarantee network connectivity and thus allow a large portion of the nodes to participate in the secure group communication, we use the system parameter relationships derived by [8] based on the phase transition theory of Erdös and Rényi for

connected random graphs. For network connectivity, we require that the neighborhood of each L2 node include 40 other nodes. This is a reasonable assumption used by [8, 5, 14]. We also need to guarantee that the L1-network (the network of L1 nodes) is connected. Using the area needed for 1000 L2 nodes where the neighborhood of each L2 node has on average 40 nodes, we use 60 L1 nodes where each L1 node is neighbor to 10–15 L1 nodes.[5] [6]

At the beginning of the simulation, each node randomly selects a key ring of size $k_A = 300$ for L1 nodes and $k_B = 150$ for L2 nodes. Nodes can establish an l-secure connection by sharing at least l keys.

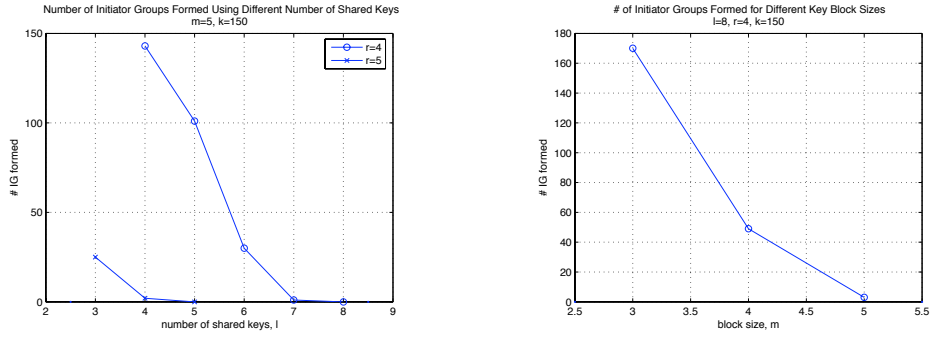
6.2 LSBS Simulation

In our simulation of LSBS protocol we exclude the L1 nodes such that our network consisted of only 1000 L2 nodes. Initiator groups (r, l)-IG are created if r neighboring nodes share l keys.

By implementing the protocol, we identified various practical concerns of IG formation which are not dealt with in [13]. In the first place, all nodes in the neighborhood of a given node u_i, may not be in transmission range. For example, although u_i might share the same l keys with neighbors u_j and u_k, it might not be able to create a $(3, l)$-IG with them because u_j and u_k are not neighbors. Therefore to make an IG, nodes must ensure that *all* potential nodes that share l keys are also neighbors. We implemented a simplified version of this condition in our simulation by only considering nodes within $R_B/2$ radius, where R_B is the transmission range of a L2 node.

It is also possible that a node belongs to more than one IG, in which case it must choose to defect to only one of the groups. We use the following defection rule in the simulation: if a node belongs to more than one IG, it defects to the IG with the larger number of members. Making the IG as large as possible has three benefits: (i) less iterations are needed in LSBS to propagate the group key to the rest of the group; (ii) more nodes contribute to the group key and therefore the formation of the key is more democratic; (iii) the disbanded IG might no longer contain enough members to create an IG and therefore we reduce the number of times the network is flooded.

Our result in Figure 5(a) show that as the number of shared keys needed to establish a secure channel decreases, a larger number of initiator groups get created. The values plotted are the average numbers obtained when the simulation is run 10 times using different seeds for the random function. For example, although for $r = 4, l = 7$ on average only two IGs are formed, there were rounds where no IG was formed. Thus to form an IG with very high probability, we must choose $l = 6$ or $l = 5$ in which case the number of IGs formed suddenly jumps to approximately 30 and 100. This means that to construct a group key, the network needs to be flooded 30 or 100 times, which is very inefficient.



(a) (b)

Fig. 5. Number of IGs created using (a) different number of shared keys l, (b) different key block sizes m.

Figure 5(b) shows the relationship between the number of IGs formed and the key block size m. Again we noticed the jump from very small number of IGs (e.g. $m = 5$) to almost 50 IGs when $m = 4$. However we know

[5] Note that we can choose a smaller density of L1 nodes than L2 nodes because the L1 nodes have a higher probability of establishing a secure connection.

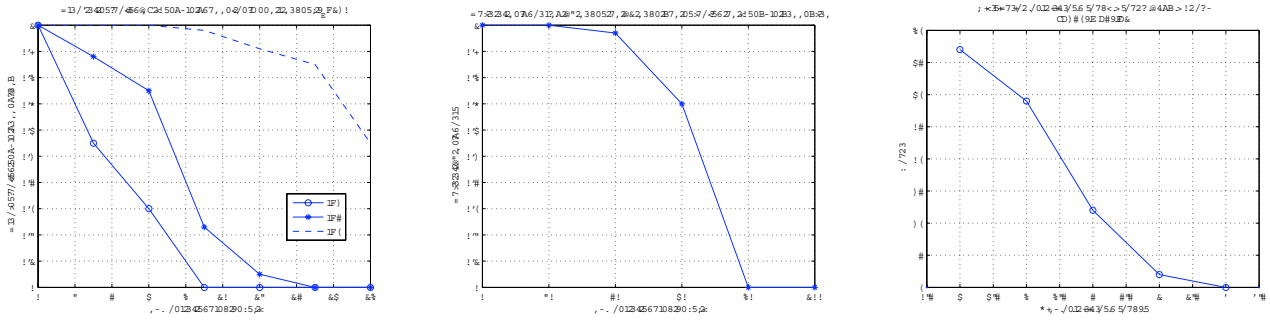
[6] We emphasize that the node density we have assumed is an overestimation of the required density in order to not result in a disconnected network. However by choosing a more refined network connectivity model or incorporating possible deployment knowledge, we can decrease network density while still maintaining connectivity. This would result in lower communication costs still.

that the larger the number of keys shared between two neighbors, the less resilient the protocol is against neighbor-compromise (see Equation 8). It is thus important to select network parameters such that allow us to minimize the number of IGs that get created but to also achieve a high degree of security against both an active and a passive adversary.

For example, if we select network parameters, $k = 150, m = 4, r = 4, l = 8$ we obtain a good balance between the number of initiator groups that are formed (around 35) as well as the resiliency of the network against an active adversary. This means that to compute the final group key, the network is flooded with 35 different partial group keys. Because flooding occurs through the secure links established between nodes, each node must perform all the computation that is required to decrypt the received partial keys (received through secure links) and encrypt then to be sent to other neighbors (secure send). Given that a node has on average 40 neighbors, it would therefore encrypt and decrypt approximately 20 times each.

6.3 LKD Performance and Discussion

By introducing hierarchy in the LSBS scheme, we are able to better control not only the formation of the local and cluster groups but also the distribution of the group keys. Figures 6(a) and (b) show the probabilities of connection for different local group sizes as well how much of the neighborhood can establish a pairwise l-secure connection with an L1 node. Our results show that with very high probability, we can achieve a connected network. In particular, an L2 node can establish a secure connection with an L1 node with very high probability. Figure 6(c) graphs the distribution of the size of the (l, r)-groups centering around each L1 node. Each group on average is made up of one L1 node and 3 L2 nodes. We emphasize that the size of a group has no influence on the security of the group key, rather it ensures a more democratic process since more nodes contribute to the calculation of the group key.



(a) One L1 node and $(r-1)$ L2 nodes (b) (c)

Fig. 6. (a) Prob. of establishing an l-secure connection between r nodes (b) Ratio of neighboring L2 nodes with which an L1 node can establish an l-secure channel. (c) Number of groups formed for different values of l.

Comparing the performance of LKD and LSBS protocols, the necessary resources of a sensing node is reduced in LKD as:

Reduced communication load. The L2-network is no longer flooded with all the partial group keys due to the clustering of the nodes and the management of the local (l, r) groups by the L1 nodes. In particular, each L2 node, with a high probability, needs to only connect to the neighboring L1 node. Furthermore if it falls in an (l, r) group, it needs to exchange $\mathcal{O}(r)$ number of messages to generate a partial cluster and group key. Therefore the number of messages that a sensing node receives and transmits is no longer a function of the neighborhood size.

Reduced computation load. LKD avoids the need for each sensing node to perform multiple decryption and re-encryptions when transporting the group key. In addition the management and decision making required for IG formation has been avoided and made a responsibility of the powerful L1 nodes. In particular in LKD with a high probability, each sensing node performs the SSKD protocol once with the neighboring L1 node. In contrast in LSBS nodes executed the SSKD protocol with every node in their neighborhood (e.g. in our simulation, this would be 40 times).

Reduced storage space. In LKD sensing nodes do not store the incidence matrix which is of the order $\mathcal{O}(k \cdot d)$ where k is the key ring size and d is the size of the neighborhood. Nodes also do not need to keep an account of the different local groups or IGs they belong to.

Finally we note that in LKD, the load on each L1 node is at most equal to the load on *every* node in LSBS. Also, the number of times LKD floods the network of L1 nodes is in the same order as the number of floods of the *whole* network for LSBS.

7 Concluding Remarks

Traditional solutions for key pre-distribution assume the existence of a single TTP. This assumption however can be very strong in practice, especially when nodes belong to different domain and they come together in an ad hoc manner, as in disaster response scenarios. In this work we showed the shortcomings of previous works [3, 13] in this area using both theoretical analysis as well as simulation. We propose a new scheme that incorporates heterogeneous nodes to ameliorate the previous shortcomings, whereby the load on resource limited nodes is reduced dramatically while in fact improving their security against node-compromising adversaries. In the course of our security analysis we pointed out a lack of security model for self-organized networks and thus presented a security model of key distribution protocols in a self-organized ad hoc network.

Our theoretical and simulation analysis pointed to a number of future research directions. The adversary model can be analyzed further, providing simulation results to compare with the theoretical results presented in this paper. We need to also come up with a good communication model to ensure that we do not end up with a disconnected graph. Finally, it is interesting to see how mobility of nodes can help ameliorate the lack of connectivity in the network.

References

1. http:www.intel.com/research/exploratory/heterogeneous.htm.
2. Mica motes. http: www.xbow.com.
3. A. Chan. Distributed symmetric key management for mobile ad hoc networks. In *Proc. of Annual Joint Conference of IEEE Computer and Communication Societies, INFOCOM*, volume 4, pages 2414–2424, March 2004.
4. H. Chan and A. Perrig. Security and privacy in sensor networks. In *IEEE Computer*, volume 36, pages 103–105, Oct 2003.
5. H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *Proc. of Symposium on Security and Privacy*, pages 11–14, May 2003.
6. W. Du, J. Deng, Y. Han, P. Varshney, J. Katz, and A. Khalili. A pariwise key predistribution scheme for wireless sensor networks. *ACM Transactions on Information and Systems Security*, 8(2):228–258, May 2005.
7. X. Du and F. Lin. Improving routing in sensor networks with heterogeneous sensor nodes. In *IEEE Vehicular Technology Conference*, pages 2528–2532, 2005.
8. L. Eschenauer and V. Gligor. A key management scheme for distributed sensor networks. In *Proc. of 9th ACM Conference on Computer and Communications Security*, pages 41–47, Washington, D.C., USA, 2002.
9. T. El Gamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, July 1985.
10. J. Lee and D. R. Stinson. A combinatorial approach to key predistribution for distributed sensor networks. In *Proc. of IEEE Wireless Communications and Networking Conference*, volume 2, pages 1200–1205, March 2005.
11. D. Liu and P. Ning. Location-based pairwise key establishments for static sensor networks. In *Proc. of 1st ACM Workshop on Security of Ad hoc and Sensor Networks*, pages 72–82, Fairfax, Virginia, USA, 2003.
12. K. Lorincz, D.J. Malan, T.R.F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. Sensor networks for emergency response: Challenges and opportunities. *IEEE Pervasive Computing*, pages 16–23, Oct-Dec 2004.
13. L. Luo, R. Safavi-Naini, J. Baek, and W. Susilo. Self-organized group key management for ad-hoc networks. In *Proc. of ACM Symposium on Information, Computer and Communications Security (AsiaCCS)*, 2006.
14. P. Traynor, H. Choi, G. Cao, S. Zhu, and T. La Porta. Establishing pair-wise keys in heterogeneous sensor networks. In *Proc. of Annual Joint Conference of IEEE Computer and Communication Societies, INFOCOM*, volume 4, pages 2414–2424, March 2006.
15. J. Wu and R. Wei. Comments on "Distributed Symmetric Key Management for Mobile Ad hoc Networks" from INFOCOM'04. Cryptology ePrint Archive, Report 2005/008, 2005. http://eprint.iacr.org/.

A Proof of Proposition 1

Proof. Let $k_A = 2k_B$, where $t = 1, s = k_B$. Therefore, each L1 node (L2 node) picks two keys (one key) from each key block. For clarity purposes, we will first concentrate on the probability of sharing a key in a single block. Later, we use this probability to calculate the probability of sharing l keys in multiple blocks and thus prove Proposition 1.

Consider two nodes, u_i and u_j, that independently pick i and j keys from a key block of size m. Let $p_{i,j}(2,l)$ denote the probability that u_i and u_j share l keys in this block, where $l \leq min(i,j)$.

If $i = j = 1$, then there are m possible choices for the key and m^2 different ways for the two nodes to select their keys. So, $p_{1,1}(2,1) = \frac{1}{m}$.

For $i = 2, j = 1$, assume the keys selected are of the form (i_1, i_2) and j_1 for nodes u_i and u_j respectively. (i_1, i_2) can be chosen in $\binom{m}{2}$ possible ways and j_1 can take on m possible values. The two nodes have a common key if $j_1 = i_1$ or $j_1 = i_2$. Now $j_1 = i_1$ for m possible values and for each i_1, i_2 takes on $(m-1)$ possible values. Therefore, the probability of having a common key in each block is:

$$p_{2,1}(2,1) = \frac{m(m-1)}{m\binom{m}{2}} = \frac{2}{m} \tag{10}$$

For $i = 2, j = 2$, assume the keys selected are of the form (i_1, i_2) and (j_1, j_2) for nodes u_i and u_j respectively. u_i and u_j will have:

- 0 shared keys: For the pair (i_1, i_2), there are $\binom{m-2}{2}$ possible pairs (j_1, j_2) with no shared keys; that is, pairs that exclude (i_1, i_2). Therefore, $p_{2,2}(2,0) = \frac{\binom{m-2}{2}}{\binom{m}{2}}$.
- 1 shared key: A given pair of the form (i_1, i_2) has i_1 common with $m-1$ pairs, and i_2 common with another $m-2$ pairs. There are $\binom{m}{2}$ distinct pairs. Therefore, $p_{2,2}(2,1) = \frac{2(m-2)}{\binom{m}{2}}$.
- 2 shared keys: For a pair (i_1, i_2) there is exactly one pair (j_1, j_2) with two collisions. There are $\binom{m}{2}$ distinct pairs. Therefore, $p_{2,2}(2,2) = \frac{1}{\binom{m}{2}}$.

We generalize the above equations for r nodes. Let $p_{X_1,...,X_r}(r,l)$ denote the probability that r nodes, where each node picks $X_1, ..., X_r$ keys respectively, all share l keys. We limit the possibilities to groups where all nodes only pick one key per block or only one node picks two keys per block and the rest pick one key. This is because each L1 node creates groups consisting of only L2 nodes.

If $X_a = 1$ where $a \in [1, r]$, then there are m possible choices for the key and m^r different ways for the two nodes to select their keys. So, $p_{1,...,1}(r,1) = \frac{1}{m^{r-1}}$.

For $X_1 = 2$ and $X_a = 1$ for $a \in [2, r]$, let x_i^a denote key i of node u_a. Then (x_1^1, x_2^1) can be chosen in $\binom{m}{2}$ possible ways and x_1^a for $a \in [2, r]$, can take on m^r possible values. The r nodes have a common key if $x_1^1 = ... = x_1^r$ or $x_2^1 = x_1^2 = ... = x_1^r$. Now $x_1^1 = ... = x_1^r$ for m possible values and for each such value, x_2^1 takes on $(m-1)$ possible values. Therefore, the probability of having a common key in each block is:

$$p_{2,1,...,1}(r,1) = \frac{m(m-1)}{m^r\binom{m}{2}} = \frac{2}{m^{r-1}} \tag{11}$$

Given the above probabilities of sharing l keys in one block for 2 nodes and r nodes, we now want to find the probability of sharing l keys in k blocks. Let $P_B(r,l)$ be the probability that r L2 nodes share l keys and $P_{A,B}(r,l)$ be the probability that $(r-1)$ L2 nodes and one L1 node share l keys. Because the probability of sharing a key in each block is independent, we can use binomial coefficients to calculate below probabilities:

$$P_B(r,l) = \binom{k}{l}\tau^l(1-\tau)^{k-l} \tag{12}$$

$$P_{A,B}(r,l) = \binom{k}{l}\lambda^l(1-\lambda)^{k-l} \tag{13}$$

where $\tau = p_{1,...,1}(r,1)$ and $\lambda = p_{2,1,...,1}(r,1)$.

Finally, we look at the special case of the probability of two L1 nodes sharing l keys, $P_{A,A}(2,l)$. Each L1 node picks 2 keys from each block. Now let α and β be non-negative integers satisfying $2\alpha + \beta = l$. To find $P_{A,A}(2,l)$ we note that two blocks can contribute 0, 1 or 2 shared keys.

This means that we have:

$$P_{A,A}(2,l) = \sum_{\substack{\alpha,\beta \\ 2\alpha+\beta=l}} \binom{k}{\alpha}\binom{k-\alpha}{\beta} p_{A,A}(2,2)^\alpha \cdot p_{A,A}(2,1)^\beta \; p_{A,A}(2,0)^{k-\alpha-\beta} \tag{14}$$

B Example of Proposition 2

In this section, we give an example of how Proposition 3 can be used to calculate the probability of r nodes sharing l keys.
Example 1: The probability of an initiating group of size r sharing exactly one key is:

$$P_{A,B}(r,1) = C_1 = \binom{s}{1} p_S(r,1)^{s-1}(1-p_S(r,1)) \cdot (1-p_{\bar{S}}(r,1))^{k-s} + (1-p_S(r,1))^s \cdot \binom{k-s}{1} p_{\bar{S}}(r,1)^{k-s-1}(1-p_{\bar{S}}(r,1))$$

Example 2: The probability of an L1 node and a L2 node sharing exactly one key. We note that $p_S(2,1) = p_{A,B}(2,1) = \frac{2}{m}$ and $p_{\bar{S}}(2,1) = p_{B,B}(2,1) = \frac{1}{m}$ as derived above. Therefore, we obtain:

$$\begin{aligned} P_{A,B}(2,1) = C_1 &= \binom{s}{1} p_S(2,1)^{s-1}(1-p_S(2,1)) \cdot (1-p_{\bar{S}}(2,1))^{k-s} + (1-p_S(2,1))^s \cdot \binom{k-s}{1} p_{\bar{S}}(2,1)^{k-s-1}(1-p_{\bar{S}}(2,1)) \\ &= s(\frac{2}{m})^{s-1}(\frac{m-2}{m}) \cdot (\frac{m-1}{m})^{k-s} + (\frac{m-2}{m})^s \cdot (k-s)(\frac{1}{m})^{k-s-1}(\frac{m-1}{m}) \end{aligned}$$